Tema 3: Jerarquía de Memorias

• Memoria Virtual

Departament d'Arquitectura de Computadors

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya





Índice

Introducción

- **Sistemas multiusuario o multiprogramado con varios programas ejecutándose concurrentemente**
  - Tamaño memoria necesario >> memoria principal
  - Sólo una pequeña porción de la memoria se está utilizando activamente en un instante determinado.
- **Los programas siempre tienen las mismas direcciones lógicas:**
  - Reubicación
  - Traducción de direcciones
- **Tamaño de un programa > memoria física**
  - Overlays
  - Memoria Virtual



Traducción de direcciones

Idea Básica:

- **Diferenciar Espacio Lógico (dirección generada por el procesador) de Espacio Físico (dirección con la que se accede a memoria).**
- **En general son diferentes**

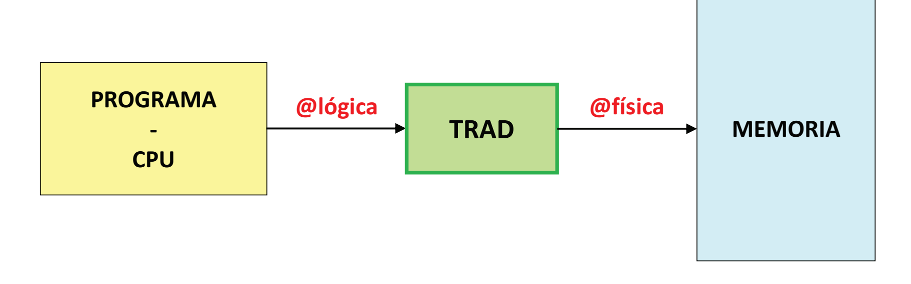
  ⇒ **Mecanismo de Traducción de Direcciones**

	Espacio lógico	Espacio físico
PDP 11/70	64 KB	256 KB
VAX-11	4 GB	32 MB

Ejemplo real (¡antiguo!)

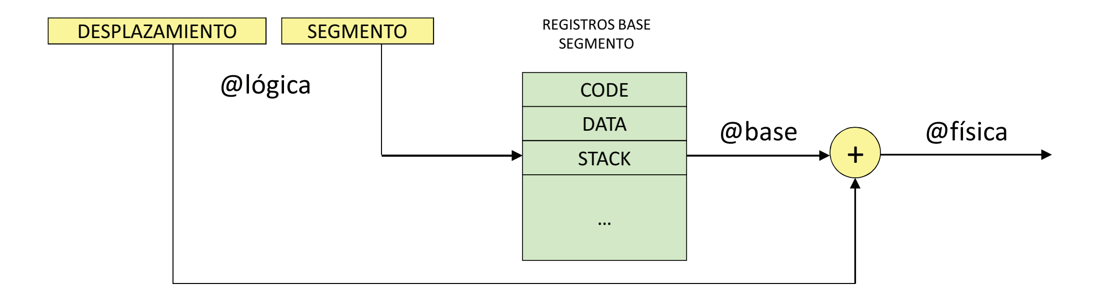
Esquemas básicos de traducción:

- **Segmentación**
- **Paginación**



Segmentación

- El programa se descompone en segmentos: código, datos, pila, …
- Cada segmento se identifica por su dirección inicial y tamaño.
- Los segmentos se almacenan de forma contigua en memoria y de forma disjunta entre segmentos.
- El mecanismo de traducción es bastante simple:



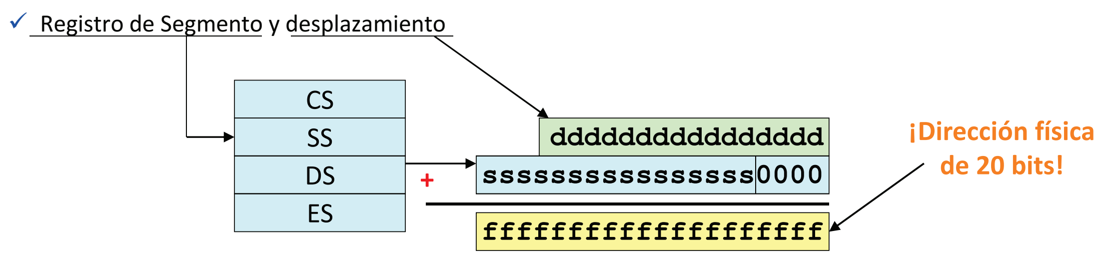
- Un cambio de contexto (usuario o programa) implica cambiar el contenido de los registros.
- Acceso lento: Acceso al banco de registros de segmentos y suma.
- Reubicación muy simple.
- Fragmentación de la memoria.
- Permite protección de los segmentos.



Segmentación: Ejemplo i8086/88

Procesador de 16 bits (bus @ de 16 bits) que generaba direcciones físicas de 20 bits.

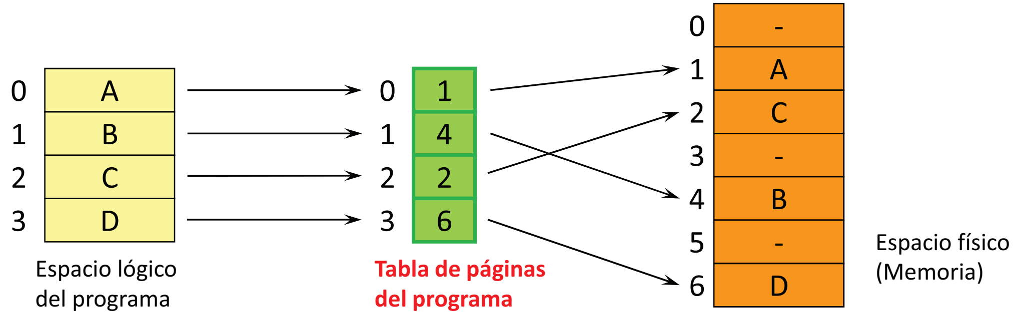
- **Disponía de 4 registros de segmento:**
  - ✓ CS: Segmento de código
  - ✓ SS: Segmento de pila
  - ✓ DS y ES: Segmentos de datos
- **Todas las direcciones se formaban con 2 componentes:**
  - ✓ Registro de Segmento y desplazamiento



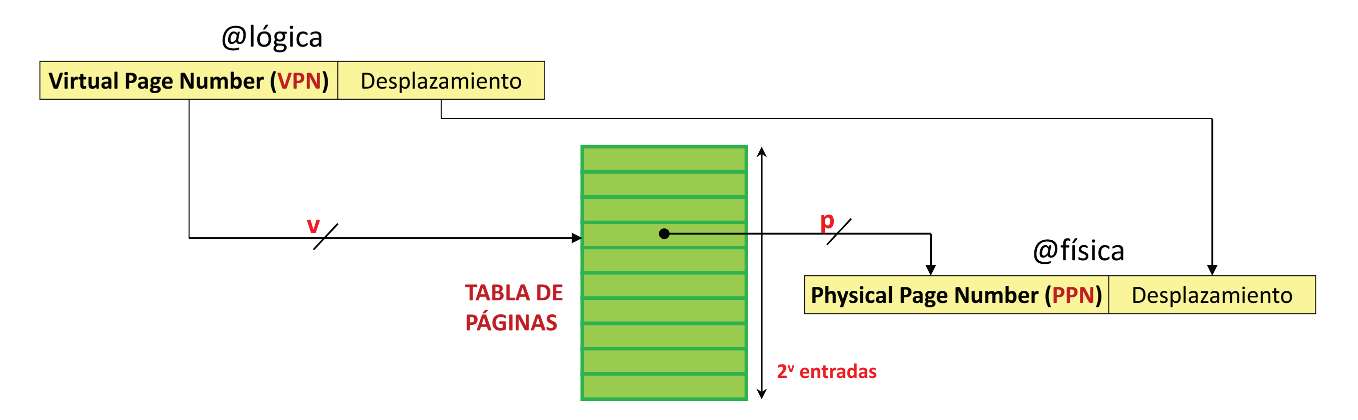
- **Cada segmento tenía un tamaño máximo de 64Kbytes.**
- **Un programa sólo podía direccionar directamente 256Kbytes, para direccionar más memoria había que cambiar el contenido de los registros de segmento.**
- **Los actuales procesadores de Intel siguen manteniendo los registros de segmento (CS, SS, DS, ES, FS y GS). En modo real funcionan igual que los antiguos i8086.**

Paginación

- El espacio lógico se divide en bloques de tamaño fijo → PÁGINAS
- Los sistemas actuales tienen páginas con tamaño entre 4 y 16 KB
- El espacio físico (MP) se divide en marcos de tamaño una página (*frames*, tramas).
- Los programas se trocean en páginas (están en disco)
- Una página puede colocarse en CUALQUIER marco de página de MP (correspondencia completamente asociativa)
- Las páginas se copian desde disco a MP cuando son referenciadas
- Hace falta una estructura de datos para saber qué hay en cada marco de página → TABLA DE PÁGINAS.



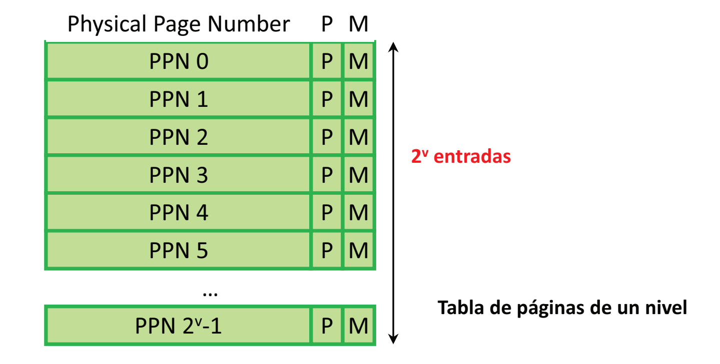
Paginación: implementación hardware



- Cálculo rápido de la dirección (no hay operaciones aritméticas).
- Fragmentación (ficheros pequeños ocupan 1 página completa).
- Reubicación muy simple.
- Permite protección de páginas.
- Página físicas y virtuales suelen tener el mismo tamaño.
- VPN y PPN pueden tener longitud diferente.
- En la mayoría de sistemas se cumple: $2^v > 2^p$.

Implementación de la Tabla de Páginas

- **Cada proceso tiene sus propias @ lógicas y físicas.**
- **Cada proceso tiene su propia Tabla de Páginas.**
- **P: bit de presencia (indica si la página está almacenada en MP).**
- **M: bit de modificación (indica si la página ha sido modificada en MP).**



Paginación: ejemplo práctico

En un sistema con direcciones virtuales de 64 bits y direcciones físicas de 43 bits, ¿Cuántos bits se necesitan para el VPN y el PPN si el tamaño de página es de 8 KB?

- $\log_2 (8 \cdot 1024) = 13$, se necesitan 13 bits para codificar el desplazamiento

@lógica 64 bits

VPN: 51 bits	Despl.: 13 bits

@física: 43 bits

PPN: 30 bits	Despl.: 13 bits

¿Qué tamaño tiene la tabla de páginas?

- **Necesitamos una entrada para cada VPN diferente y en cada entrada necesitamos un mínimo de 30 bits para codificar la PPN**

⇒Tamaño mínimo $2^{51} \cdot 30$ bits = $7{,}5 \cdot 2^{50}$ bytes = 7,5 PB (¡Peta bytes!)

¡Un poco GRANDE!, ¿NO?

Implementación de la Tabla de Páginas

- **En un procesador actual, la TP sería mucho más grande que la Memoria Principal.**
- **Solución: Tablas de Páginas de múltiples niveles (no las estudiaremos)**
  - **Sólo una parte de la tabla de páginas está en MP**
  - **Se requieren varios accesos a la tabla de páginas para conocer la @física de la página**
- **En este curso utilizaremos como modelo una Tabla de Páginas de un solo nivel almacenada siempre en MP.**

- **La Tabla de Páginas es accedida en cada referencia a memoria**
- **Si la Tabla de Páginas es de un nivel y se almacena en Memoria Principal**
  - 1 acceso a MP necesita 1 acceso a la Tabla de Páginas y 1 acceso al dato

⇒MUY LENTO

- **Solución: Tener una memoria cache "especial" para la tabla de páginas**
  - **Translation Lookaside Buffer (TLB) – Buffer de traducción anticipada**



En el TLB no hay datos (o programas), sólo información para acelerar la traducción de direcciones.



Traducción de direcciones con TLB

Translation Lookaside Buffer (TLB)

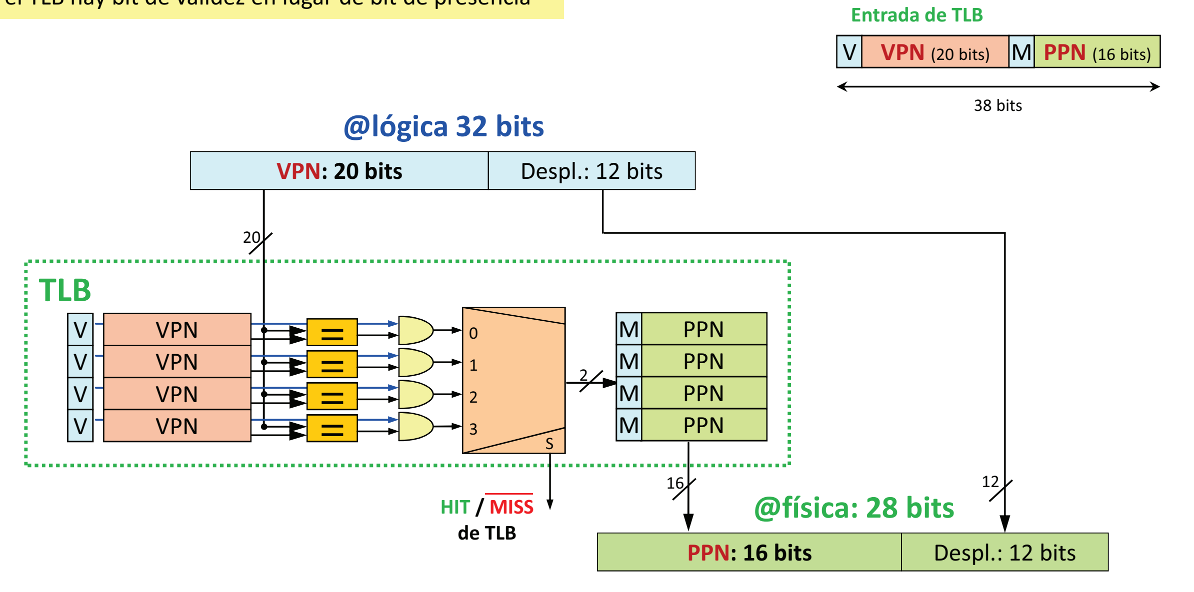
- Sirve para acelerar el proceso de traducción de direcciones
- Tiene una estructura similar (campos) a la Tabla de Páginas
- Sólo guarda algunas de las entradas de la TP
- Contiene más entradas de página que las páginas que caben en la cache L1 (contiene traducciones de datos residentes en L2 y en MP).
- Procesadores de mediados de los 90 tenían TLB de 128 entradas, 32-64 KB de cache L1 y páginas de 4-16KB

Características principales:

- Integrado en el mismo chip que en el procesador
- Pocas entradas (64-128) (1 entrada por página)
- Completamente asociativo
- Tasa de fallos muy baja
- Muy rápido (debido a que tiene pocas entradas de pocos bits)
- Algoritmo de reemplazo (LRU, PseudoLRU, FIFO, Random, ..)

Traducción de direcciones con TLB

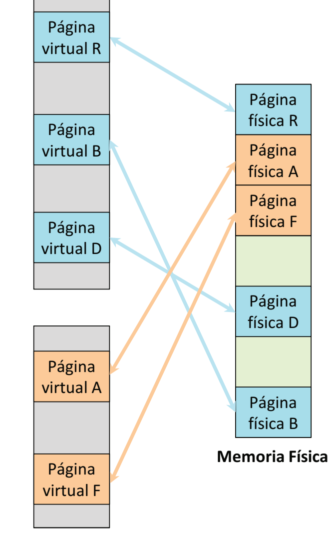
- 32 bits de dirección lógica
- 28 bits de dirección física
- Páginas de 4KB (2^{12} bytes)
- TLB de 4 entradas
- En el TLB hay bit de validez en lugar de bit de presencia





Paginación: protección

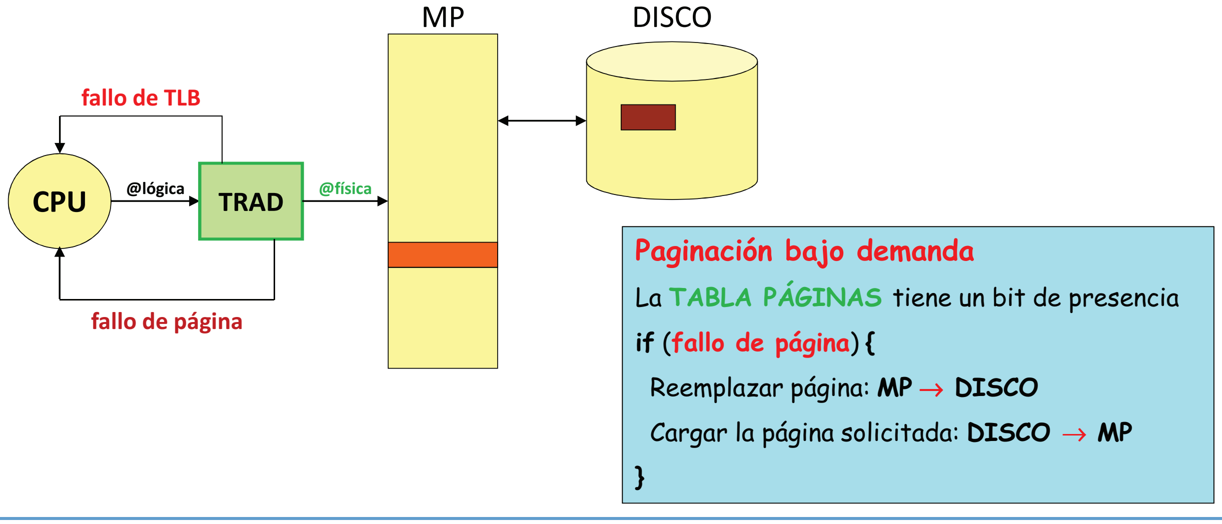
- **Cada proceso tiene su propia Tabla de Páginas**
- **Ventajas**
  - Los procesos comparten espacio físico de direcciones, pero tienen espacios virtuales distintos
  - El sistema de traducción de direcciones asegura que las páginas virtuales de cada proceso se mapean en páginas físicas distintas (en MP y en disco)
  - Si dos procesos quieren compartir sus datos, algunos SO permiten realizar una petición específica para que algunas de sus direcciones virtuales se asignen a las mismas direcciones físicas
- **Inconvenientes**
  - El mapeo de direcciones virtuales a físicas es parte del estado del proceso
  - Cuando el SO realiza un cambio de contexto, hay que invalidar el TLB
  - Cuando se comienza a ejecutar un proceso hay muchos fallos de TLB
  - Para solventar este problema, algunos sistemas actuales incorporan un identificador de proceso en el TLB (coexisten entradas de procesos distintos)



Memoria virtual

La Memoria Virtual permite:

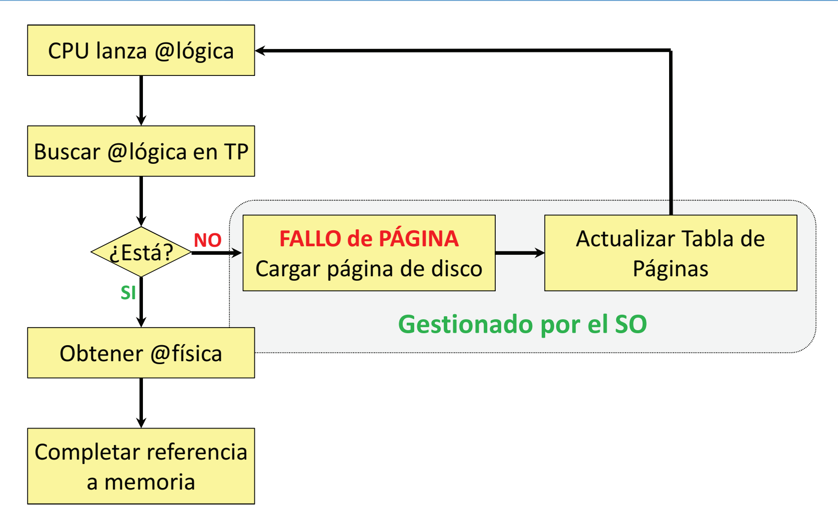
- Ejecutar un programa con espacio lógico > espacio físico.
- Ejecutar un programa parcialmente cargado en Memoria.
- Proteger el espacio de direcciones de los programas de ser accedido por otros programas



Memoria virtual

- **¿Quién gestiona la memoria virtual?**
  - El Sistema Operativo (software)
  - ¿Porqué no el hardware?
- **¿Cuándo se trae una página de Disco a MP?**
  - Bajo demanda en caso de fallo (hay otros modelos)
- **¿Dónde se ubica una página en MP?**
  - En cualquier marco, política totalmente asociativa
- **¿Qué página de la MP se substituye en caso de fallo?**
  - Algoritmos de reemplazo muy sofisticados. La tasa de fallos es MUY importante. Un fallo puede costar millones de ciclos porque hay que acceder a disco. La decisión es software y hay mucho tiempo para tomarla.
  - Las páginas modificadas hay que escribirlas en disco.
  - Tasa de fallos: 0,00001% - 0,001%
- **¿Qué se hace con las escrituras?**
  - COPY BACK + WRITE ALLOCATE

Paginación bajo demanda (sin TLB)



Espacio virtual del proceso

Página virtual R

Página virtual B

Página virtual D

Memoria Física

Página física R

Página física D

Página disco R

Página disco D

Página disco B

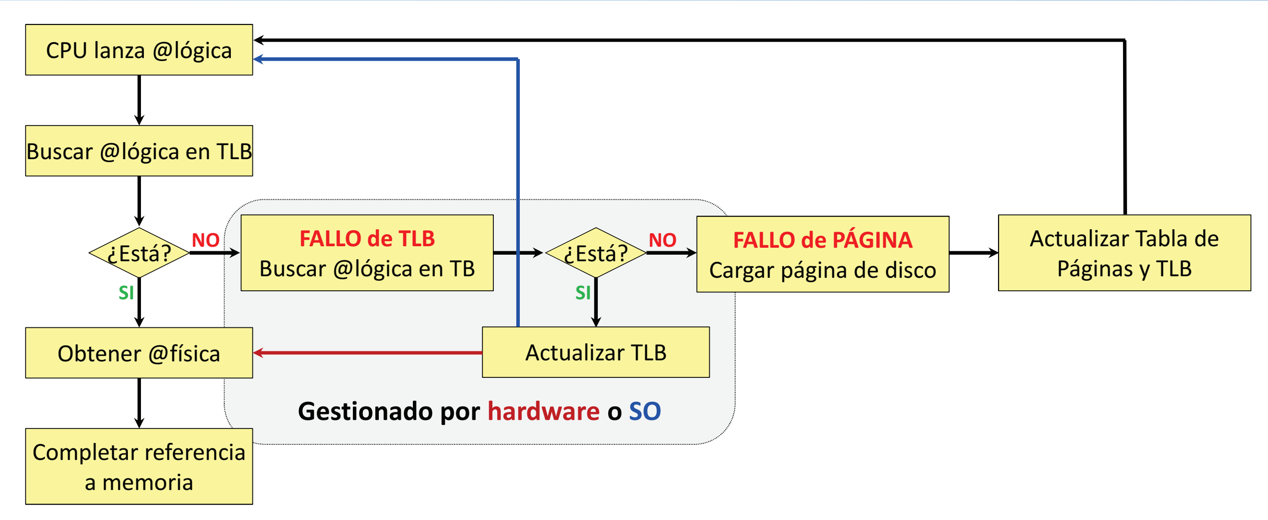
DISCO DURO

Página no presente en MP

Fallo de página

- Necesita acceder al disco (milisegundos).
- Se resuelve mediante una excepción. Puede tardar varios millones de ciclos.
- Los SO suelen aprovechar un fallo de página para cambiar de contexto
- Cuando se reinicia el proceso, después de resolver el fallo de página, se **reejecuta la instrucción** y se repite el acceso a memoria.

Paginación bajo demanda (con TLB)



Fallo de TLB

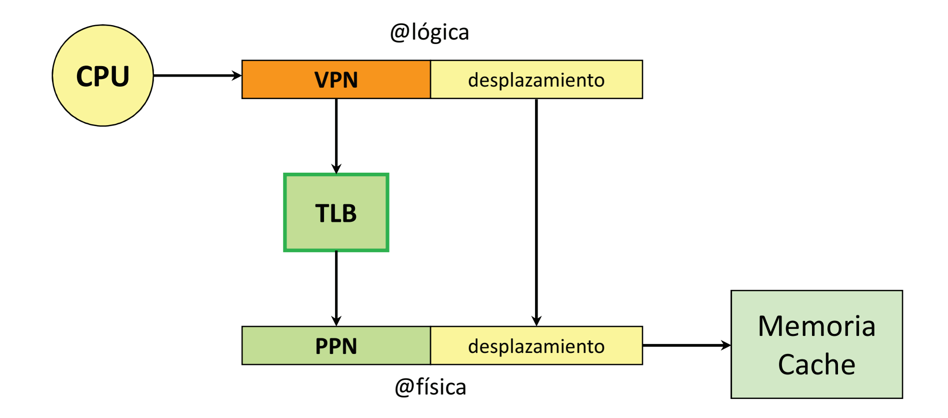
- Se puede resolver mediante una excepción o incluso por hardware.
- Requiere un tiempo relativamente corto para ser resuelto si la página está en la Tabla de Páginas
- Típicamente, se resuelve en unos centenares de ciclos

Juntando Memoria Virtual y Memoria Cache

- **La traducción de direcciones y la memoria cache son conceptos ortogonales:**
  - La memoria cache permite acelerar los accesos a memoria
  - La traducción de direcciones permite soportar memoria virtual
    - ✓ El TLB es sólo un mecanismo de aceleración del proceso de traducción
- **Un sistema puede tener sólo memoria cache, sólo traducción de direcciones, ambos mecanismos o ninguno de ellos**
- **Los actuales procesadores de propósito general cuentan con una jerarquía de uno o más niveles de cache y mecanismos de traducción de direcciones con el correspondiente TLB**
- **En este último caso, ¿cuándo se efectúa la traducción de direcciones lógicas a físicas, antes o después de acceder a la Memoria Cache?**
- **Tres posibilidades:**
  - Traducción **antes** de acceder a Memoria Cache
  - Traducción **después** de acceder a Memoria Cache
  - Traducción y acceso a Memoria Cache **simultáneos**

Juntando Memoria Virtual y Memoria Cache

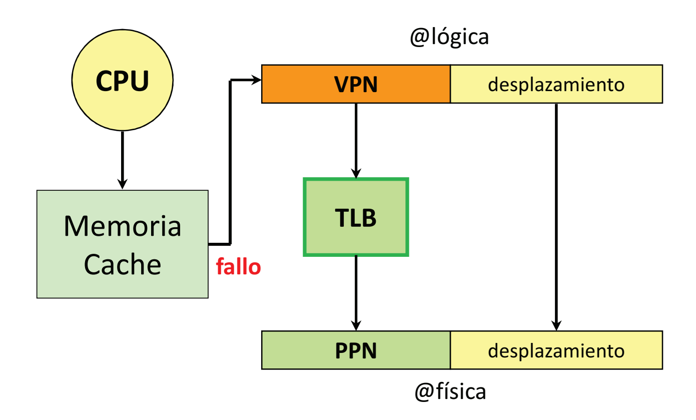
- **Traducción antes de acceder a Memoria Cache**



- **Memoria Cache de direcciones físicas**
- **Lento: un acceso a memoria necesita un acceso TLB + acceso MC**

Juntando Memoria Virtual y Memoria Cache

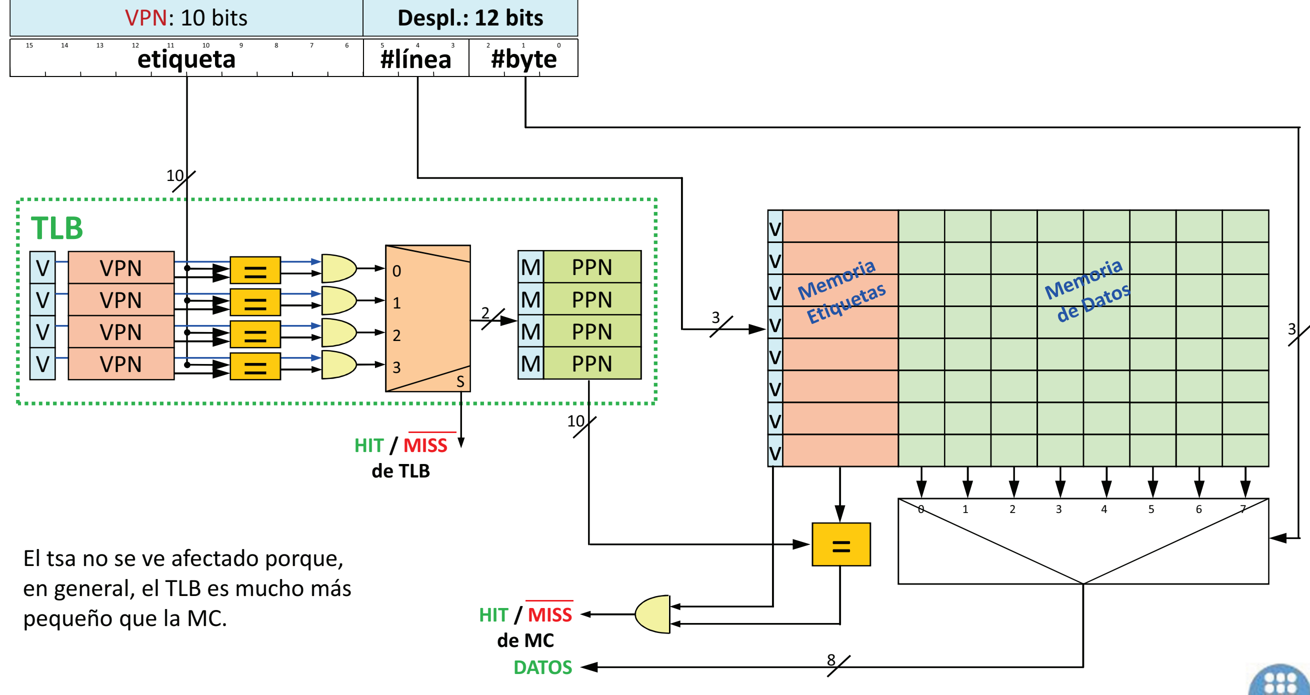
- **Traducción después de acceder a Memoria Cache**



- **Memoria Cache de direcciones lógicas**
- **Se realiza traducción SÓLO en caso de fallo en MC**
- **Aumenta el coste de un fallo de MC**

Juntando Memoria Virtual y Memoria Cache

- **Traducción en TLB y acceso a Memoria Cache simultáneos**



El tsa no se ve afectado porque, en general, el TLB es mucho más pequeño que la MC.

Juntando Memoria Virtual y Memoria Cache

- **Traducción en TLB y acceso a Memoria Cache simultáneos**
  - Se busca en la MC con la parte de la dirección que corresponde al desplazamiento (línea y byte de la línea)
  - La memoria de etiquetas contiene etiquetas **FÍSICAS**
  - Se traduce únicamente la **página LÓGICA** que corresponde a la etiqueta y se comprueba si la línea de la MC es la línea física buscada.
  - Restringe el tamaño de la Memoria Cache:
    - ✓ #conjuntos · tamaño línea $\leq$ tamaño página

VPN	Desplazamiento (tamaño página)

	#Conjunto o #línea MC	#byte (tam. Línea)